Table of Contents

Resolved

Jobs Alternate Between Running and Queued States

***Problem**: Users experience unexplained job behavior, where a job alternates between Running and Queued states.*

Status: Resolved

If you are still experiencing issues with this problem, contact the NAS Control room: (800) 331-8737, (650) 604-4444, support@nas.nasa.gov.

Actions:

Updated 06.02.11 - NAS systems staff have implementing an automated method to detect processes that are running out of memory. The owner of the job will get an e-mail and the job will be terminated or blocked from rerunning.

Tips:

- If your job is bouncing between Running and Queued states in PBS, then you should assume you have an out-of-memory (OOM) situation and kill your job using the command *qdel*. You can get confirmation of the OOM situation by checking whether a job was killed by the OOM killer; or contact the NAS Control Room staff at (800) 331-8737 or (650) 604-4444.
- If you have processes running out of memory, you can increase the memory available to the processes. For example:
    - When running on Harpertown nodes, try running on Westmeres, which have twice as much memory per core.
    - When running on Westmere nodes, try running on Nehalems, which have 50% more memory per core.
    - Try running with fewer active cores in each node, and running on more nodes.
    - Run the *rank0* process in a node by itself, and add 1 to the number of nodes.

Background:

The way in which the system kills processes that are running out of memory has been changed. While the new method leaves the host node in a better state than before, the user no longer gets a message that the out-of-memory condition occurred. Furthermore, the killing is so "efficient" that PBS does not get notified. Consequently, PBS re-queues the job as if it were affected by a system problem.

In addition, SUSE Linux Enterprise 11 (SLES11) has slightly less memory available for processes than was available under SLES10. The combination means that *some* codes that ran fine with SLES10 could fail inexplicably with SLES11.

Files Fail to Open

***Problem**: Users experience errors opening or inquiring about existing files using Intel Fortran on Lustre filesystems.*

Status: Resolved

If you are still experiencing issues related to this problem, contact the NAS Control room: (800) 331-8737, (650) 604-4444, support@nas.nasa.gov.

Actions:

Updated 06.14.11 - A kernel patch was installed and tested, and NAS systems staff verified that the problem no longer occurs. The patch was implemented during the Pleiades dedicated time June 8-13.

Tips:

Several workarounds were available before the kernel patch was installed.

1. Pre-load a getcwdHack library before running the executable.

   This library is available under the directory */nasa/lustre_getcwd* and was built under SLES11 SP1.

   If you plan to use this library under SLES10, you can copy the directory

   /nasa/lustre_getcwd

   to your own directory, and build it under SLES10 (bridge1 & bridge2).

   Note that no modification to your source code is needed.

   Add the following to your PBS script before running your executable.

   For csh:

   ```
   setenv LD_PRELOAD /nasa/lustre_getcwd/libgetcwdHack.so
   ```

For bash:

export LD_PRELOAD=/nasa/lustre_getcwd/libgetcwdHack.so

2. Modify your source code to re-try the file open. For example:

```
integer open_stat  (needs to be declared in this routine)

      ntries = 0   ! number of tries to open the file

 100  OPEN (UNIT=10, FILE='some_filename', STATUS='old', IOSTAT=open_stat)

      if (open_stat .ne. 0) then
         ntries = ntries + 1
         if (ntries .gt. 10) then
            print *, 'Cannot open file some_filename'
            call MPI _ABORT(MPI_COMM_WORLD, 1, ierr)
         endif
         call sleep(1)   ! to wait 1 second before retrying
         go to 100
      endif
```

3. If the file is intended to be read-only, you can change the file permission by typing "chmod 400 *filename*" or you can modify the source code to specify that the file is read-only.

```
OPEN (UNIT=10, FILE='some_filename', STATUS='old', ACTION='READ')
```

Background

With Intel Fortran and Lustre filesystems, a problem has been reported when large numbers of MPI ranks attempt to open the same file, resulting in a variety of error messages:

- forrtl: severe (9): permission to access file denied, unit xx, file /filename
- forrtl: severe (29): file not found, unit xx, file /filename
- forrtl: No such file or directory
  forrtl: severe (29): file not found, unit xx, file -/filename

In these cases, a superfluous backslash ("/") or an additional random character, such as hyphen ("-") was placed in front of the filename by the Fortran Runtime Library. This is because a *getcwd* command was issued to find the current directory and it gets "bad" information from the system. This results in the file being inaccessible or not found.

A Lustre bug report proposes a kernel patch, as well as a library built to pre-load in which *getcwd* is retried several times.